

本节内容

图

基本操作

知识总览

图的基本操作:

- $\text{Adjacent}(G, x, y)$: 判断图 G 是否存在边 $\langle x, y \rangle$ 或 (x, y) 。
- $\text{Neighbors}(G, x)$: 列出图 G 中与结点 x 邻接的边。
- $\text{InsertVertex}(G, x)$: 在图 G 中插入顶点 x 。
- $\text{DeleteVertex}(G, x)$: 从图 G 中删除顶点 x 。
- $\text{AddEdge}(G, x, y)$: 若无向边 (x, y) 或有向边 $\langle x, y \rangle$ 不存在, 则向图 G 中添加该边。
- $\text{RemoveEdge}(G, x, y)$: 若无向边 (x, y) 或有向边 $\langle x, y \rangle$ 存在, 则从图 G 中删除该边。
- $\text{FirstNeighbor}(G, x)$: 求图 G 中顶点 x 的第一个邻接点, 若有则返回顶点号。若 x 没有邻接点或图中不存在 x , 则返回-1。
- $\text{NextNeighbor}(G, x, y)$: 假设图 G 中顶点 y 是顶点 x 的一个邻接点, 返回除 y 之外顶点 x 的下一个邻接点的顶点号, 若 y 是 x 的最后一个邻接点, 则返回-1。
- $\text{Get_edge_value}(G, x, y)$: 获取图 G 中边 (x, y) 或 $\langle x, y \rangle$ 对应的权值。
- $\text{Set_edge_value}(G, x, y, v)$: 设置图 G 中边 (x, y) 或 $\langle x, y \rangle$ 对应的权值为 v 。

图的存储

邻接矩阵

邻接表

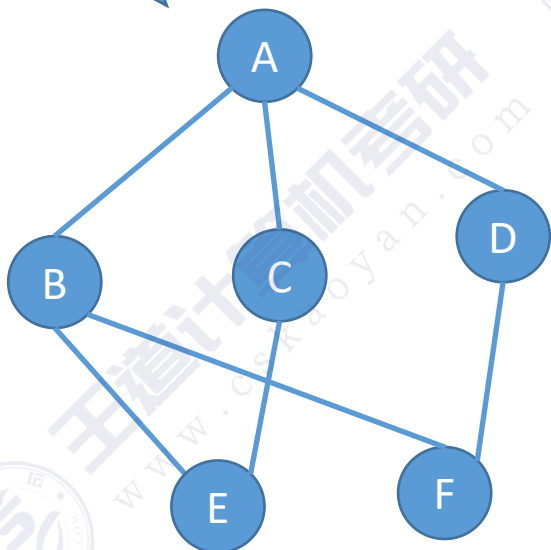
十字链表

邻接多重表

图的基本操作

- $\text{Adjacent}(G,x,y)$: 判断图 G 是否存在边 $\langle x, y \rangle$ 或 (x, y) 。

无向图



邻接矩阵

$O(1)$

	data		A	B	C	D	E	F
0	A	A	0	1	1	1	0	0
1	B	B	1	0	0	0	1	1
2	C	C	1	0	0	0	1	0
3	D	D	1	0	0	0	0	1
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

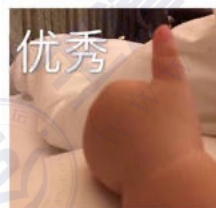
邻接表

$O(1) \sim O(|V|)$

	data	*first
0	A	→ [1] → [2] → [3] ^
1	B	→ [0] → [4] → [5] ^
2	C	→ [0] → [4] ^
3	D	→ [0] → [5] ^
4	E	→ [1] → [2] ^
5	F	→ [1] → [3] ^

图的基本操作

- Adjacent(G,x,y): 判断图G是否存在边<x, y>或(x, y)。



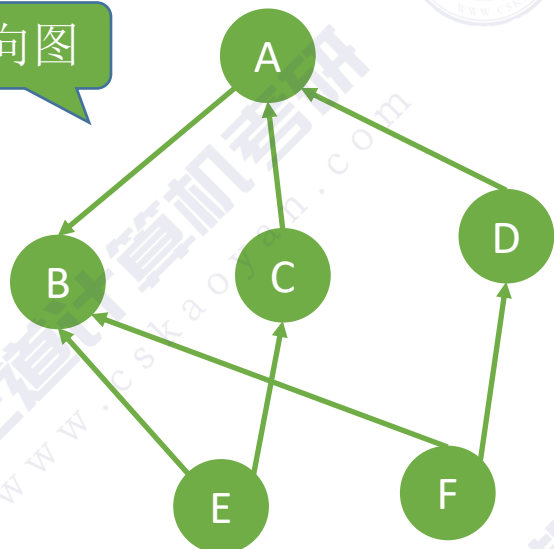
邻接矩阵

$O(1)$

邻接表

$O(1) \sim O(|V|)$

有向图



	data		A	B	C	D	E	F
0	A	A	0	1	0	0	0	0
1	B	B	0	0	0	0	0	0
2	C	C	1	0	0	0	0	0
3	D	D	1	0	0	0	0	0
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

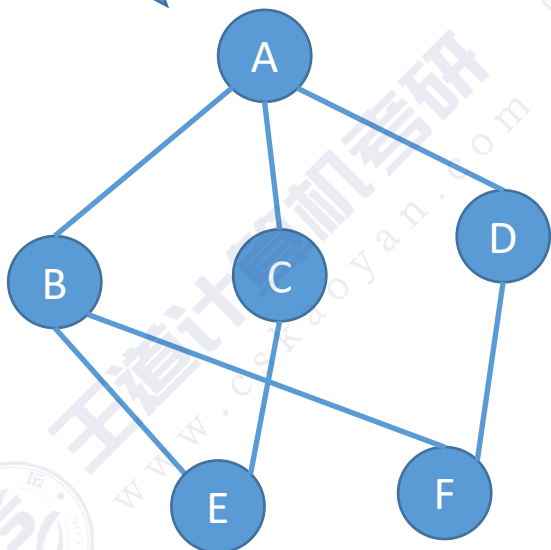
	data	*first
0	A	→ 1 ^
1	B	→ ^
2	C	→ 0 ^
3	D	→ 0 ^
4	E	→ 1 → 2 ^
5	F	→ 1 → 3 ^

图的基本操作

- Neighbors(G,x): 列出图G中与结点x邻接的边。



无向图



邻接矩阵

$O(|V|)$

	data		A	B	C	D	E	F
0	A	A	0	1	1	1	0	0
1	B	B	1	0	0	0	1	1
2	C	C	1	0	0	0	1	0
3	D	D	1	0	0	0	0	1
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

邻接表

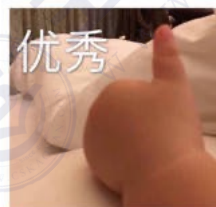
$O(1) \sim O(|V|)$

	data	*first
0	A	→ [1] → [2] → [3] ^
1	B	→ [0] → [4] → [5] ^
2	C	→ [0] → [4] ^
3	D	→ [0] → [5] ^
4	E	→ [1] → [2] ^
5	F	→ [1] → [3] ^

图的基本操作

- Neighbors(G,x): 列出图G中与结点x邻接的边。

万一是个稀疏图呢?



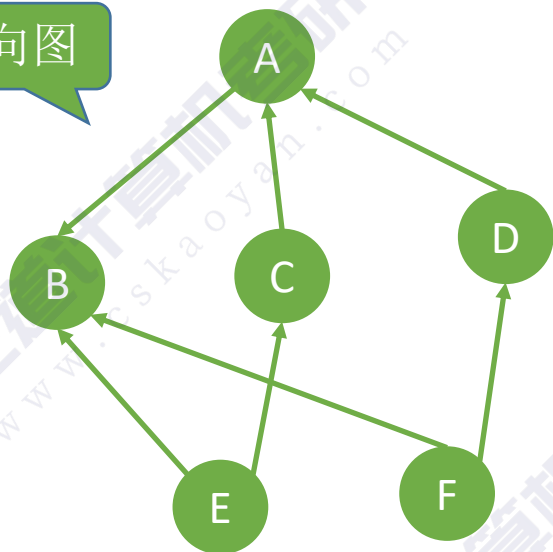
邻接矩阵

$O(|V|)$

邻接表

出边: $O(1) \sim O(|V|)$
入边: $O(|E|)$

有向图



	data		A	B	C	D	E	F
0	A	A	0	1	0	0	0	0
1	B	B	0	0	0	0	0	0
2	C	C	1	0	0	0	0	0
3	D	D	1	0	0	0	0	0
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

	data	*first
0	A	→ 1 ^
1	B	→ ^
2	C	→ 0 ^
3	D	→ 0 ^
4	E	→ 1 → 2 ^
5	F	→ 1 → 3 ^

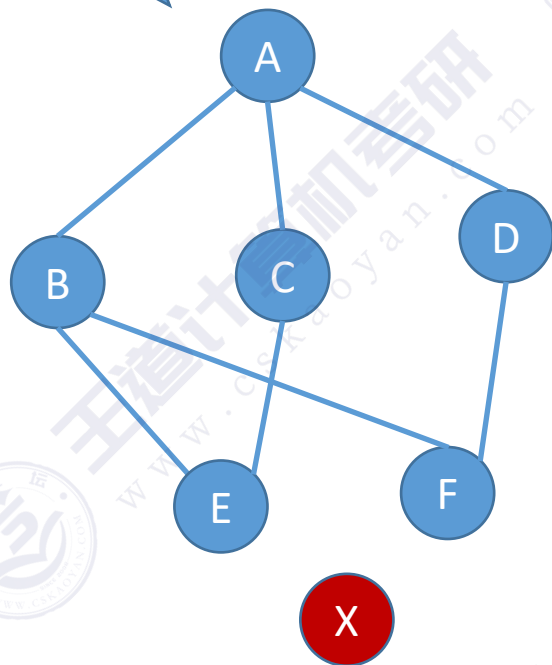
图的基本操作

- InsertVertex(G,x): 在图G中插入顶点x。



有向图也类似

无向图



邻接矩阵

$O(1)$

	data
0	A
1	B
2	C
3	D
4	E
5	F
6	X

	A	B	C	D	E	F	X
A	0	1	1	1	0	0	0
B	1	0	0	0	1	1	0
C	1	0	0	0	1	0	0
D	1	0	0	0	0	1	0
E	0	1	1	0	0	0	0
F	0	1	0	1	0	0	0
X	0	0	0	0	0	0	0

邻接表

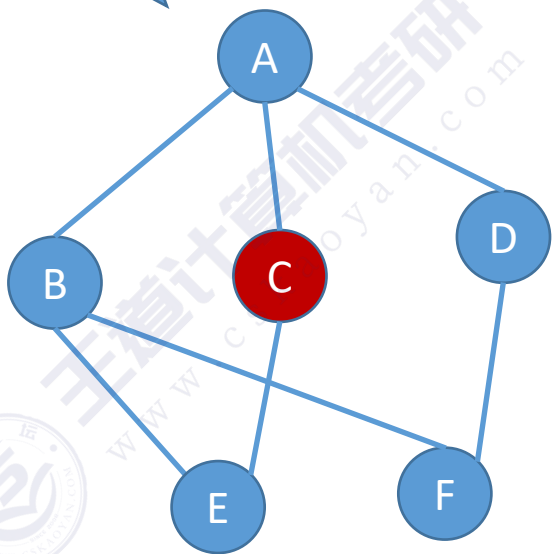
$O(1)$

	data	*first
0	A	1 → 2 → 3 ^
1	B	0 → 4 → 5 ^
2	C	0 → 4 ^
3	D	0 → 5 ^
4	E	1 → 2 ^
5	F	1 → 3 ^
6	X	^

图的基本操作

- DeleteVertex(G, x): 从图 G 中删除顶点 x 。

无向图



邻接矩阵

	data		A	B	C	D	E	F
0	A	A	0	1	1	1	0	0
1	B	B	1	0	0	0	1	1
2	C	C	1	0	0	0	1	0
3	D	D	1	0	0	0	0	1
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

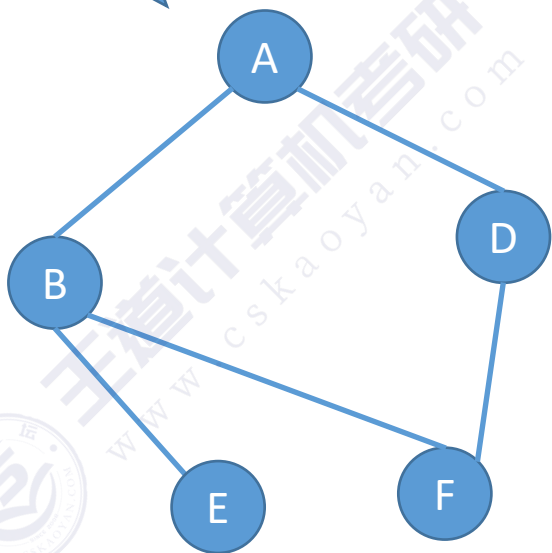
邻接表

	data	*first
0	A	→ [1] → [2] → [3] → ^
1	B	→ [0] → [4] → [5] → ^
2	C	→ [0] → [4] → ^
3	D	→ [0] → [5] → ^
4	E	→ [1] → [2] → ^
5	F	→ [1] → [3] → ^

图的基本操作

- DeleteVertex(G, x): 从图 G 中删除顶点 x 。

无向图



邻接矩阵

$O(|V|)$

	data		A	B	空	D	E	F
0	A	A	0	1	0	1	0	0
1	B	B	1	0	0	0	1	1
2	空	空	0	0	0	0	0	0
3	D	D	1	0	0	0	0	1
4	E	E	0	1	0	0	0	0
5	F	F	0	1	0	1	0	0

邻接表

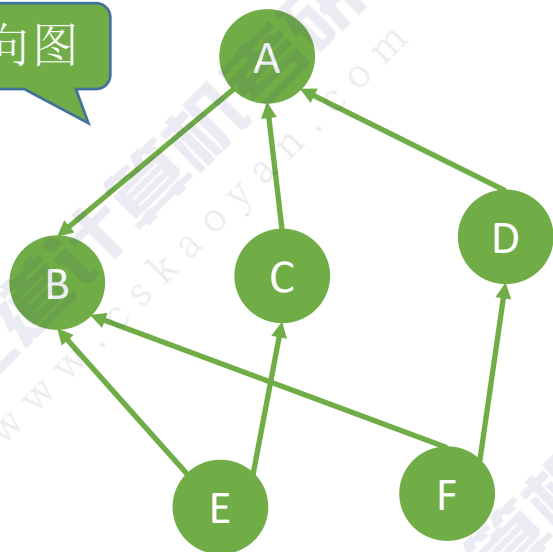
$O(1) \sim O(|E|)$

	data	*first
0	A	1 → 3 ^
1	B	0 → 4 → 5 ^
2	空	^
3	D	0 → 5 ^
4	E	1 ^
5	F	1 → 3 ^

图的基本操作

- DeleteVertex(G,x): 从图G中删除顶点x。

有向图



邻接矩阵

$O(|V|)$

	data		A	B	C	D	E	F
0	A	A	0	1	0	0	0	0
1	B	B	0	0	0	0	0	0
2	C	C	1	0	0	0	0	0
3	D	D	1	0	0	0	0	0
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

邻接表

	data	*first
0	A	→ 1 ^
1	B	→ ^
2	C	→ 0 ^
3	D	→ 0 ^
4	E	→ 1 → 2 ^
5	F	→ 1 → 3 ^

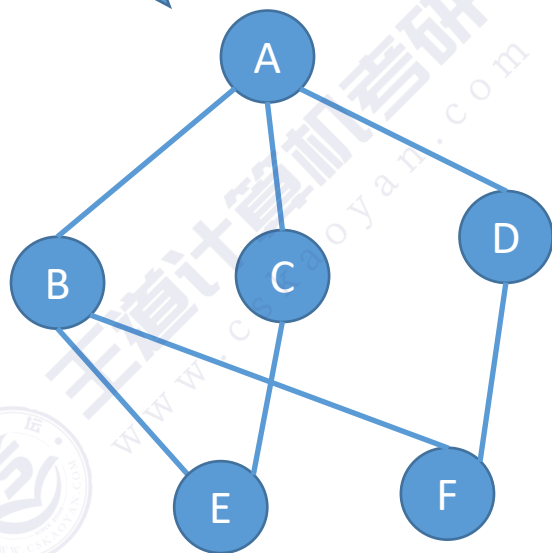
删出边: $O(1) \sim O(|V|)$
删入边: $O(|E|)$

图的基本操作

- $\text{AddEdge}(G, x, y)$: 若无向边 (x, y) 或有向边 $\langle x, y \rangle$ 不存在, 则向图 G 中添加该边。



无向图



邻接矩阵

$O(1)$

	data		A	B	C	D	E	F
0	A	A	0	1	1	1	0	0
1	B	B	1	0	0	0	1	1
2	C	C	1	0	0	0	1	0
3	D	D	1	0	0	0	0	1
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

邻接表

$O(1)$

	data	*first
0	A	1 → 2 → 3 ^
1	B	0 → 4 → 5 ^
2	C	0 → 4 ^
3	D	0 → 5 ^
4	E	1 → 2 ^
5	F	1 → 3 ^

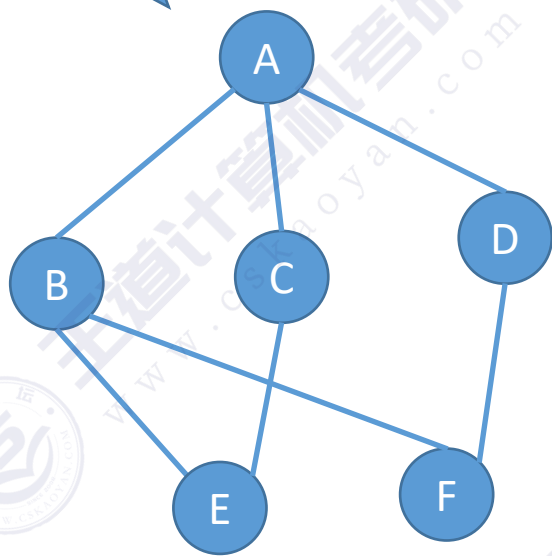
图的基本操作

- `RemoveEdge(G,x,y)`: 若无向边 (x,y) 或有向边 $\langle x,y \rangle$ 存在, 则从图 G 中删除该边。



有向图
也类似

无向图



邻接矩阵

$O(1)$

	data		A	B	C	D	E	F
0	A	A	0	1	1	1	0	0
1	B	B	1	0	0	0	1	1
2	C	C	1	0	0	0	1	0
3	D	D	1	0	0	0	0	1
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

邻接表

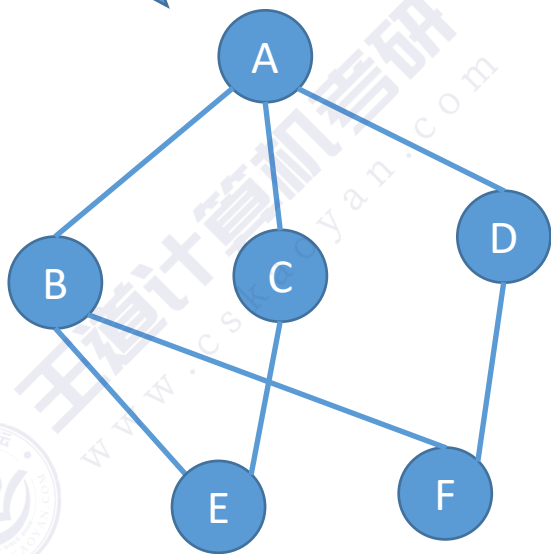
$O(1) \sim O(|V|)$

	data	*first
0	A	1 → 2 → 3 ^
1	B	0 → 4 → 5 ^
2	C	0 → 4 ^
3	D	0 → 5 ^
4	E	1 → 2 ^
5	F	1 → 3 ^

图的基本操作

- **FirstNeighbor(G,x)**: 求图G中顶点x的第一个邻接点, 若有则返回顶点号。若x没有邻接点或图中不存在x, 则返回-1。

无向图



邻接矩阵

$O(1) \sim O(|V|)$

	data
0	A
1	B
2	C
3	D
4	E
5	F

	A	B	C	D	E	F
A	0	1	1	1	0	0
B	1	0	0	0	1	1
C	1	0	0	0	1	0
D	1	0	0	0	0	1
E	0	1	1	0	0	0
F	0	1	0	1	0	0

邻接表

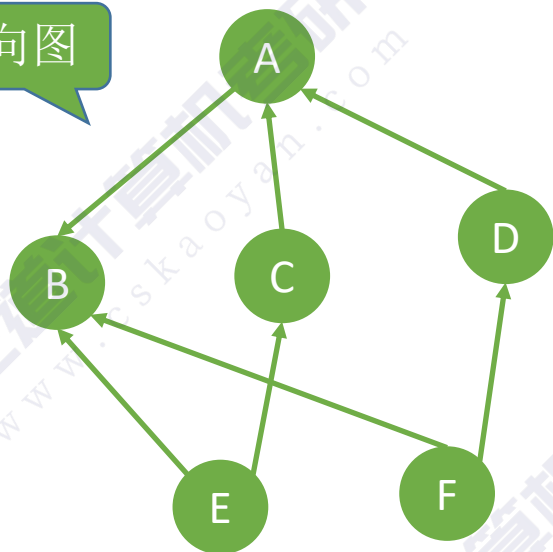
$O(1)$

	data	*first
0	A	→ 1 → 2 → 3 ^
1	B	→ 0 → 4 → 5 ^
2	C	→ 0 → 4 ^
3	D	→ 0 → 5 ^
4	E	→ 1 → 2 ^
5	F	→ 1 → 3 ^

图的基本操作

- **FirstNeighbor(G,x)**: 求图G中顶点x的第一个邻接点, 若有则返回顶点号。若x没有邻接点或图中不存在x, 则返回-1。

有向图



邻接矩阵

$O(1) \sim O(|V|)$

	data		A	B	C	D	E	F
0	A	A	0	1	0	0	0	0
1	B	B	0	0	0	0	0	0
2	C	C	1	0	0	0	0	0
3	D	D	1	0	0	0	0	0
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

邻接表

找出边邻接点: $O(1)$

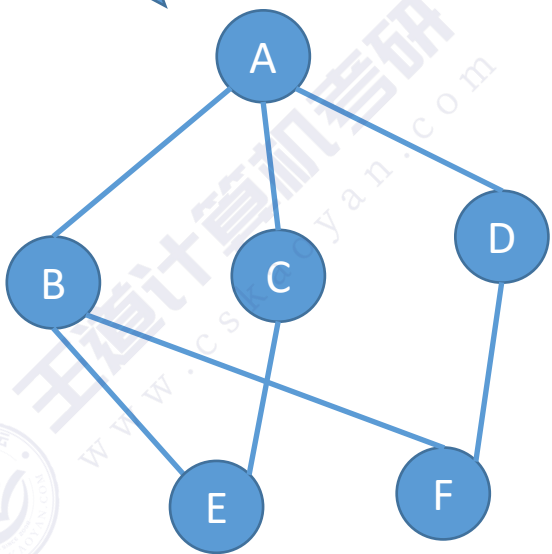
找入边邻接点: $O(1) \sim O(|E|)$

	data	*first
0	A	→ 1 ^
1	B	→ ^
2	C	→ 0 ^
3	D	→ 0 ^
4	E	→ 1 → 2 ^
5	F	→ 1 → 3 ^

图的基本操作

- **NextNeighbor(G, x, y):** 假设图 G 中顶点 y 是顶点 x 的一个邻接点, 返回除 y 之外顶点 x 的下一个邻接点的顶点号, 若 y 是 x 的最后一个邻接点, 则返回-1。

无向图



邻接矩阵

$O(1) \sim O(|V|)$

	data
0	A
1	B
2	C
3	D
4	E
5	F

	A	B	C	D	E	F
A	0	1	1	1	0	0
B	1	0	0	0	1	1
C	1	0	0	0	1	0
D	1	0	0	0	0	1
E	0	1	1	0	0	0
F	0	1	0	1	0	0

邻接表

$O(1)$

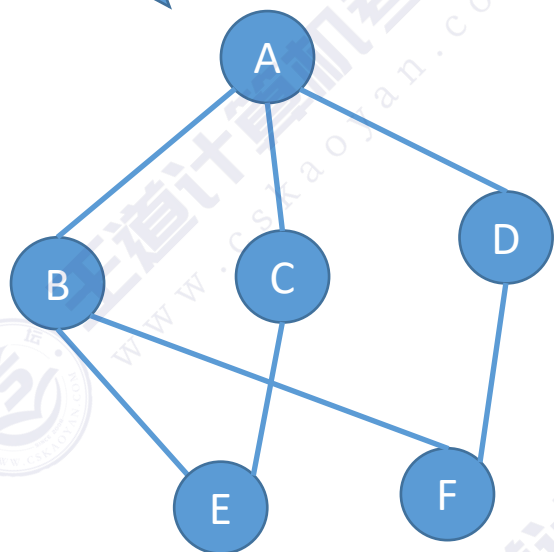
	data	*first
0	A	→ [1] → [2] → [3 ^]
1	B	→ [0] → [4] → [5 ^]
2	C	→ [0] → [4 ^]
3	D	→ [0] → [5 ^]
4	E	→ [1] → [2 ^]
5	F	→ [1] → [3 ^]

图的基本操作

- $\text{Get_edge_value}(G,x,y)$: 获取图 G 中边 (x,y) 或 $\langle x,y \rangle$ 对应的权值。
- $\text{Set_edge_value}(G,x,y,v)$: 设置图 G 中边 (x,y) 或 $\langle x,y \rangle$ 对应的权值为 v 。
- $\text{Adjacent}(G,x,y)$: 判断图 G 是否存在边 $\langle x,y \rangle$ 或 (x,y) 。

雷同，核心在于找到边

无向图



邻接矩阵

$O(1)$

	data		A	B	C	D	E	F
0	A	A	0	1	1	1	0	0
1	B	B	1	0	0	0	1	1
2	C	C	1	0	0	0	1	0
3	D	D	1	0	0	0	0	1
4	E	E	0	1	1	0	0	0
5	F	F	0	1	0	1	0	0

邻接表

$O(1) \sim O(|V|)$

	data	*first
0	A	→ 1 → 2 → 3 ^
1	B	→ 0 → 4 → 5 ^
2	C	→ 0 → 4 ^
3	D	→ 0 → 5 ^
4	E	→ 1 → 2 ^
5	F	→ 1 → 3 ^

知识回顾与重要考点

- $\text{Adjacent}(G,x,y)$: 判断图 G 是否存在边 $\langle x, y \rangle$ 或 (x, y) 。
- $\text{Neighbors}(G,x)$: 列出图 G 中与结点 x 邻接的边。
- $\text{InsertVertex}(G,x)$: 在图 G 中插入顶点 x 。
- $\text{DeleteVertex}(G,x)$: 从图 G 中删除顶点 x 。
- $\text{AddEdge}(G,x,y)$: 若无向边 (x, y) 或有向边 $\langle x, y \rangle$ 不存在, 则向图 G 中添加该边。
- $\text{RemoveEdge}(G,x,y)$: 若无向边 (x, y) 或有向边 $\langle x, y \rangle$ 存在, 则从图 G 中删除该边。
- $\text{FirstNeighbor}(G,x)$: 求图 G 中顶点 x 的第一个邻接点, 若有则返回顶点号。若 x 没有邻接点或图中不存在 x , 则返回-1。
- $\text{NextNeighbor}(G,x,y)$: 假设图 G 中顶点 y 是顶点 x 的一个邻接点, 返回除 y 之外顶点 x 的下一个邻接点的顶点号, 若 y 是 x 的最后一个邻接点, 则返回-1。
- $\text{Get_edge_value}(G,x,y)$: 获取图 G 中边 (x, y) 或 $\langle x, y \rangle$ 对应的权值。
- $\text{Set_edge_value}(G,x,y,v)$: 设置图 G 中边 (x, y) 或 $\langle x, y \rangle$ 对应的权值为 v 。

此外, 还有图的遍历算法, 包括深度优先遍历和广度优先遍历。